CLAIMS

What is claimed is:

1. An apparatus for detecting and stopping intrusions, DOS attacks, and computer worm comprising of:

   a) Means to dynamically load information about vulnerabilities and exposures

   b) Means to intercept application data

   c) Means to detect exploitation of vulnerabilities and exposures in application data

   d) Means to take custom action to stop the exploitation of vulnerabilities and exposures

2. An apparatus of claim 1 further comprising of:

   a) Virtual patches that contain information about vulnerabilities and exposures

   b) Instances of interpreter core engine that will dynamically load virtual patches for vulnerabilities and exposures, detect their exploitation, and take action to stop it.

   c) Virtual proxies that interface with transport layer to intercept application data and manage application sessions, and interface with instances of interpreter core engine to detect exploitation of vulnerabilities and exposures

3. An apparatus of claim 2, wherein component (a), a virtual patch, further comprising of:

a) Information about vulnerability/exposure detailed enough to detect its exploitation and take necessary action to stop it

b) Action required when an exploitation of a vulnerability/exposure is detected

4. An apparatus of claim 2, wherein component (b), an interpreter core engine, further comprising of:

a) A different Interpreter Configuration Structure (ICS) for every application to control the processing of application data for the purpose of detecting exploitation of vulnerabilities and exposures

b) Separate Decoder-Plugin for every application to provide decoding procedures for application information elements and can be loaded dynamically

c) An External Interpreter Configuration (EIC) to capture the virtual patches for vulnerabilities and exposures in an application

e) An interface that allows the virtual proxy to invoke this instance of interpreter core engine to check for the exploitation of vulnerabilities and exposures

f) An interface to dynamically load virtual patches (DP AND EIC) and build ICS from DP and EIC

5. An apparatus of claim 2, wherein component (b), a virtual proxy, further comprising of one or more of the following components:

a) An interface with transport layer to intercept application data and forward the modified data or the same data to the original destination

b) An interface that an instance of the interpreter engine uses to send same or modified application data to original destination or send a custom response to the sender of the data that contains exploitation of vulnerabilities and exposures

c) Session management

d) An interface to fine tune generic DOS control parameters

---

6. An apparatus of claim 4, wherein component (a), an Interpreter Configuration Structure (ICS), further comprising of one or more of the following components:

a) Semantic tree that contains information on how and when to trigger the virtual patch processing procedures for various vulnerabilities and exposures

b) Procedures to control session state

c) Protocol level parameters that control decoder plug-in

d) Additional protocol level parameters that control processing of vulnerabilities, and exposures

e) List of application information elements that decoder plug-in should extract.

f) Procedures used to process events when exceptions are detected by the decoder

g) Procedures used to initialize a session context

h) Procedures to free-up storage for a session context

i) Procedure to free-up memory when the interpreter configuration data structure is removed

j) Compiled set of regular expressions, pattern lists, and value list

k) Generic DOS control parameters

l) A reference count

m) Static semantic elements that are used to build a semantic tree

---

7. An apparatus of claim 4, wherein component (b), a decoder-plugin (DP), further comprising of:

a) A procedure to build semantic trees

b) Procedures to control session state

c) Protocol level parameters that control decoder plug-in

d) Procedures used to process events when exceptions are detected by the decoder

e) A procedure to initialize decoder related session context

f) A procedure to free decoder related session context

g) A procedure to create a data structure which contains all information elements that can be enabled for decoding

h) Procedures for decoding information elements and maintaining session context related to decoder session context

8. An apparatus of claim 4, wherein component (c), an External Interpreter Configuration (EIC), further comprising of one or more of the following components:

a) Procedure to enhance or change semantic trees

b) Procedure to change protocol level parameters that control decoder plug-in

c) Protocol level parameters that control processing of vulnerabilities and exposures

d) List of application information elements that decoder plug-in should extract.

e) Changes to the procedures used to process events when exceptions are detected by the decoder

f) Procedure used to initialize session context related to processing of vulnerabilities, and exposures

g) Procedure used to free external session context

h) Virtual patch processing procedures for vulnerabilities and exposures

i) All regular expressions, pattern lists, and value lists that need to be compiled Generic DOS control parameters

9. A method for detecting and stopping intrusions, DOS attacks, and computer worms comprising the steps of:

a) Dynamically loading the information about known vulnerabilities and exposures

b) Intercepting application data

c) Processing application data using the information about vulnerabilities and exposures to detect and stop intrusions, DOS attacks, and computer worms

10. A method of claim 9 further comprising the steps of:

a) Instantiating one virtual proxy for every application that needs protection

b) Instantiating one interpreter core engine for every application that needs protection

c) Interfacing of a virtual proxy with the transport layer

d) Interfacing of a virtual proxy with an instance of interpreter core engine

e) Decoding application information elements

f) Executing virtual patch processing procedures of known vulnerabilities and exposures

g) Dynamically loading virtual patches for vulnerabilities and exposures

11. A method of claim 10, wherein step (c), interfacing of a virtual proxy with the transport layer, further comprising steps of:

a) Intercepting application data information

b) Sending application information elements to original destination

c) Instructing transport layer to drop, pass, or bypass application information

12. A method of claim 10, wherein step (d), interfacing of an instance of interpreter core engine with the virtual proxy further comprising steps of:

a) Virtual proxy passing new application data to the interpreter instance

b) The interpreter instance instructing its virtual proxy to pass, drop, or change an information element

c) The instance instructing its virtual proxy to insert a new information element

d) The instance instructing its virtual proxy to bypass or remove application data without passing it to the instance

13. A method of claim 10, wherein step (e), decoding application information elements, further comprising steps of decoding in a manner to optimize overall performance of the method:

a) An instance of interpreter core engine using ICS to decide what information elements should be decoded and what should be skipped

b) Using a procedure provided by DP to perform decoding

14. A method of claim 11, wherein step (f), further comprising steps of:

a) Using the static semantic element of an information element to decide what virtual patch processing procedures to trigger

b) Executing selected virtual patch processing procedures

c) Using the virtual proxy interface to optimize the performance of a virtual patch action

15. A method of claim 12, wherein step (g) further comprising steps of:

a) Converting virtual patches for an application into EIC and DP

b) Dynamically loading EIC and DP for an application and building ICS from them

c) Updating internal data to ensure new application sessions use new ICS

16. An apparatus for reducing time it takes to capture information about vulnerabilities and exposures for the purpose of detecting and stopping intrusions, computer worms, and DOS attacks comprising of:

a) Custom language constructs to capture information about vulnerabilities and exposures

b) A translator for converting custom language constructs into an intermediate form that a security system can read to detect and stop the exploitation of vulnerabilities and exposures

17. An apparatus of claim 15 further comprising of:

a) High level language constructs to capture decoding procedures for application information elements

b) High level language constructs to capture processing procedures for vulnerabilities and exposures

c) High level language constructs to capture protocol context

d) High level language constructs to capture session state context

e) A translator for converting a high-level virtual patch description into EIC and DP that the interpreter core engine can read

18. An apparatus of claim 17, wherein component (a) further comprising of:

a) EBNF-like construct to define decoding procedures

b) 3-G Language-like construct to define decoding procedures

19. An apparatus of claim 17, wherein component (b) further comprising of:

a) Validation Constructs

b) Action Constructs

20. An apparatus of claim 19, wherein component (a), validation constructs, further comprising one or more of the following components:

a) Stateful or stateless information element-length check:

b) Stateful or stateless information element matching with regular expression:

c) Stateful or stateless information element value check:

d) Stateful or stateless information element value list check:

e) Stateful or stateless information element character set check

f) Stateful or stateless information element check for tagged components:

g) Stateful or stateless information element check for invalid-content:

h) Stateful parameter validation:

i) Generic DOS control

j) 3G Language such as C or C++ Like Validation procedure

21. An apparatus of claim 19, wherein component (b), action constructs, further comprising one or more of the following components:

a) A construct that makes it easy to send alert and log messages,

b) A construct that makes it easy to configure how to record audit trail

c) A construct that allows removing malicious content from the session

d) A construct that helps in normalizing malicious content with acceptable content

e) A construct that helps in inserting new content at specific mark in an information element

f) Application specific response construct

22. A method for reducing time it takes to capture information about vulnerabilities and exposures for the purpose of detecting and stopping their exploitation comprising steps of:

a) Using custom language constructs to capture the information about vulnerabilities and exposures that allows the detection and stopping of their exploitation

b) Using a translator to convert this information into an intermediate form that a security system can dynamically read, and use it to detect and stop the exploitation of vulnerabilities and exposures.

23. A method of claim 22 further comprising steps of:

a) Using high level decode-constructs to define decoding procedures for information elements that are needed for processing of virtual patches
b) Using high level constructs for session state context to define new state parameters needed for processing of virtual patches
c) Using high level constructs for protocol context parameters to capture new protocol level parameters and changes to the existing one
d) Using high level constructs to capture the semantics of the vulnerabilities and exposures.
e) Using a translator to convert the high representation of virtual patches into EIC and DP

24. An apparatus to convert information about vulnerabilities and exposures into an intermediate form that optimizes the processing speed of the method and apparatus for stopping intrusions, DOS attacks, and computer worms comprising of:

a) Means to selectively enable and generate decoding procedure for information elements
b) Means to convert high level description of information about vulnerabilities and exposures in an intermediate form that optimizes loading, flushing, and passing of information elements

c) Means to extract regular expressions matches, pattern matches, value list comparisons and convert them in an intermediate form that optimizes their processing

d) Means to generate a structure that permits selective processing of vulnerabilities and exposures

25. A method to convert information about vulnerabilities and exposures into an intermediate form that optimizes the processing speed of the method and apparatus for stopping intrusions, DOS attacks, and computer worms comprising steps of:

a) Selectively enabling decoding and generating decoding procedure for information elements

b) Converting high level description of information about vulnerabilities and exposures in an intermediate form that optimizes loading, flushing, and passing of information elements

c) Extracting regular expressions matches, pattern matches, value list comparisons and convert them in an intermediate form that optimizes their processing

d) Generating a structure that permits selective processing of vulnerabilities and exposures

26. A method of claim 25 further comprising steps of:

a) Generating information element decoding procedures from high level constructs

b) Generating information needed for EIC translation

c) Generating EIC from high level constructs

27. A method of claim 26, wherein step (a), further comprising steps of:

a) Generating decoding procedures from EBNF construct

b) Generating decoding procedures from 3G like constructs

28. A method of claim 26, wherein step (c), further comprising steps of:

a) Generating Procedure to enhance or change semantic trees for selectively processing virtual patches

b) Generating Virtual Patch Processing Procedures

c) Generating Procedures to manage protocol level and session level contexts

29. A method of claim 28, wherein step (b), further comprising steps of:

a) Generating Variable Definition

b) Generating statements to retrieve all dependencies

c) Generating statements to process vulnerabilities and exposures

d) Generating statements for passing a select set of decoded information elements

e) Generating statements for loading a select set of decoded information elements

f) Generating statements for flushing a select set of decoded information elements

g) Extracting regular expressions matches, pattern matches, value list comparisons and convert them in an intermediate form that optimizes their processing